AI ASSISTED CODING

NAME: N.RUKMINI

ENROLL NUMBER: 2403A52402

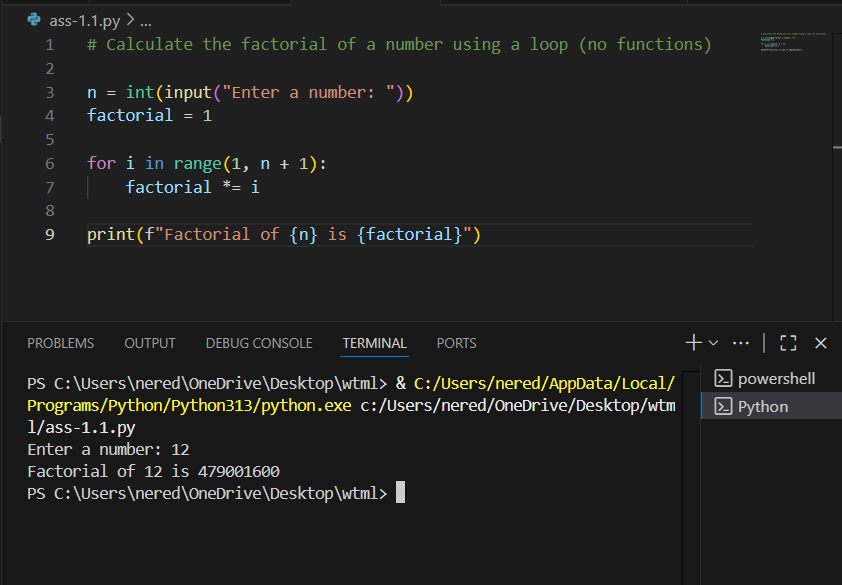
BATCH NUMBER :14

Lab assignment-1.1

Prompt 1: Factorial without Functions

Use GitHub Copilot to generate a Python program that calculates the  
factorial of a number without defining any functions (using loops  
directly in the main code)

Code(screenshot):



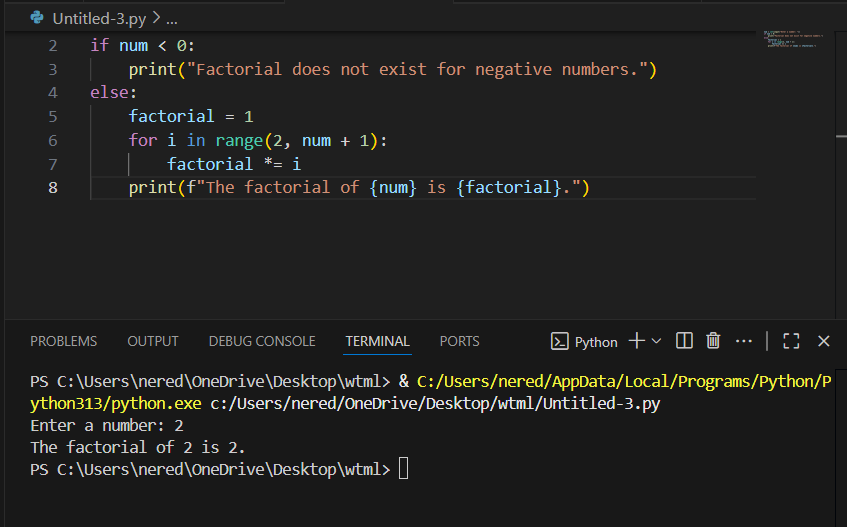
Code explanation:

This code calculates the factorial of a user-provided number using a loop:

* It prompts the user to enter a number and stores it in n.
* It initializes factorial to 1.
* It uses a for loop from 1 to n, multiplying factorial by each number in the range.
* After the loop, it prints the result, which is the factorial of the input number.

Prompt 2: Improving Efficiency  
• Description:  
Examine the Copilot-generated code from Task 1 and demonstrate  
how its efficiency can be improved (e.g., removing unnecessary  
variables, optimizing loops).

Code(screen shot):



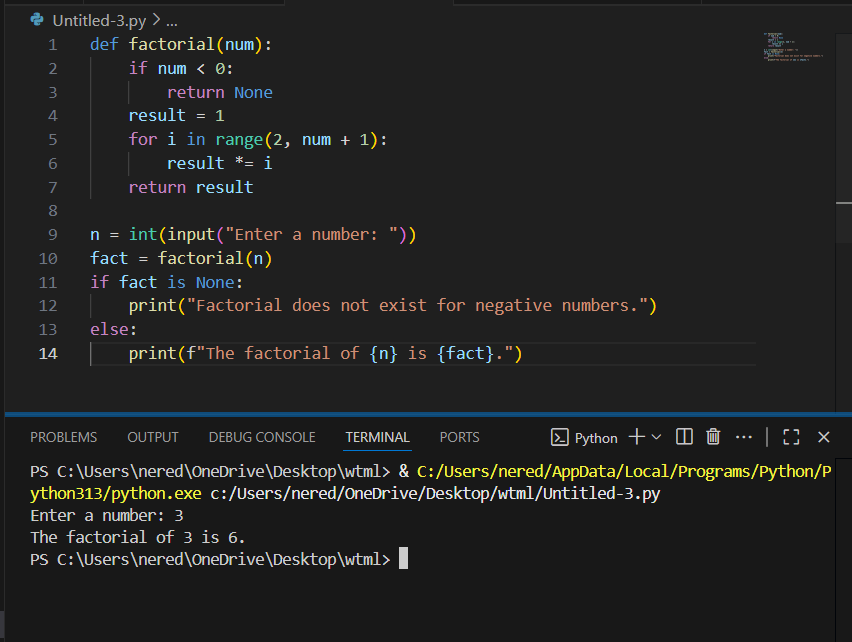
Code explanation:

This code calculates the factorial of a number entered by the user:

* [num = int(input("Enter a number: "))](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)  
  Prompts the user to enter a number and converts the input to an integer.
* [if num < 0:](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)  
  Checks if the number is negative.
* [print("Factorial does not exist for negative numbers.")](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)  
  If negative, prints a message since factorials are only defined for non-negative integers.
* else:  
  If the number is zero or positive:
  + [factorial = 1](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)  
    Initializes the factorial result to 1.
  + [for i in range(2, num + 1):](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)  
    Loops from 2 up to and including [num](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
    - [factorial \*= i](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)  
      Multiplies [factorial](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) by each value of [i](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html" \o ") in the loop.
  + [print(f"The factorial of {num} is {factorial}.")](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)  
    Prints the final factorial value

prompt 3: Factorial with Functions  
Use GitHub Copilot to generate a Python program that calculates the  
factorial of a number using a user-defined function.

Code(screen shot):



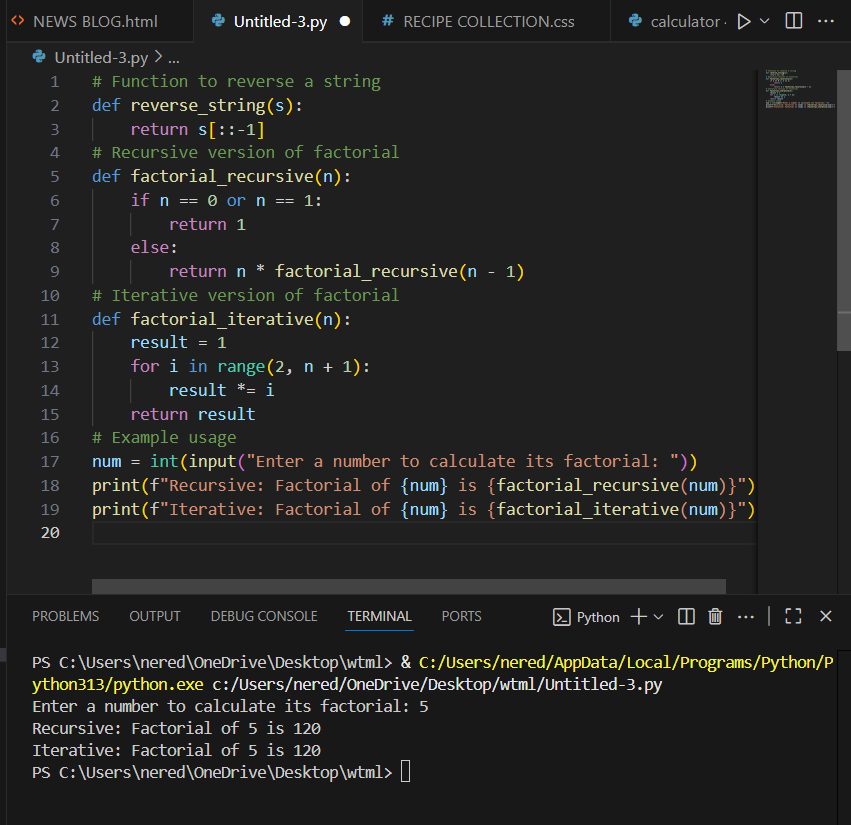
Code explanation:

This program calculates the factorial of a number using a user-defined function:

* The [factorial(num)](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) function checks if the input is negative. If so, it returns None.
* If the input is zero or positive, it initializes [result](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) to 1 and multiplies it by each integer from 2 up to [num](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
* The main code gets a number from the user, calls the [factorial](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) function, and stores the result.
* If the result is None, it prints a message for negative numbers. Otherwise, it prints the factorial value.

Prompt 4: Comparative Analysis – With vs Without Functions  
Differentiate between the Copilot-generated factorial program with functions and without functions in terms of logic, reusability, and execution

Code(screen shot):



Code explanation:

* **reverse\_string(s):**  
  This function takes a string [s](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) and returns its reverse using slicing ([s[::-1]](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)).
* **factorial\_recursive(n):**  
  This function calculates the factorial of [n](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) recursively.
  + If [n](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) is 0 or 1, it returns 1 (base case).
  + Otherwise, it returns [n \* factorial\_recursive(n - 1)](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
* **factorial\_iterative(n):**  
  This function calculates the factorial of [n](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) using a loop.
  + It initializes [result](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) to 1.
  + Then multiplies [result](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) by each number from 2 to [n](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
* **Example usage:**
  + The user is prompted to enter a number.
  + The program prints the factorial of that number using both the recursive and iterative functions.

Prompt 5:Iterative vs Recursive Factorial

● Description:

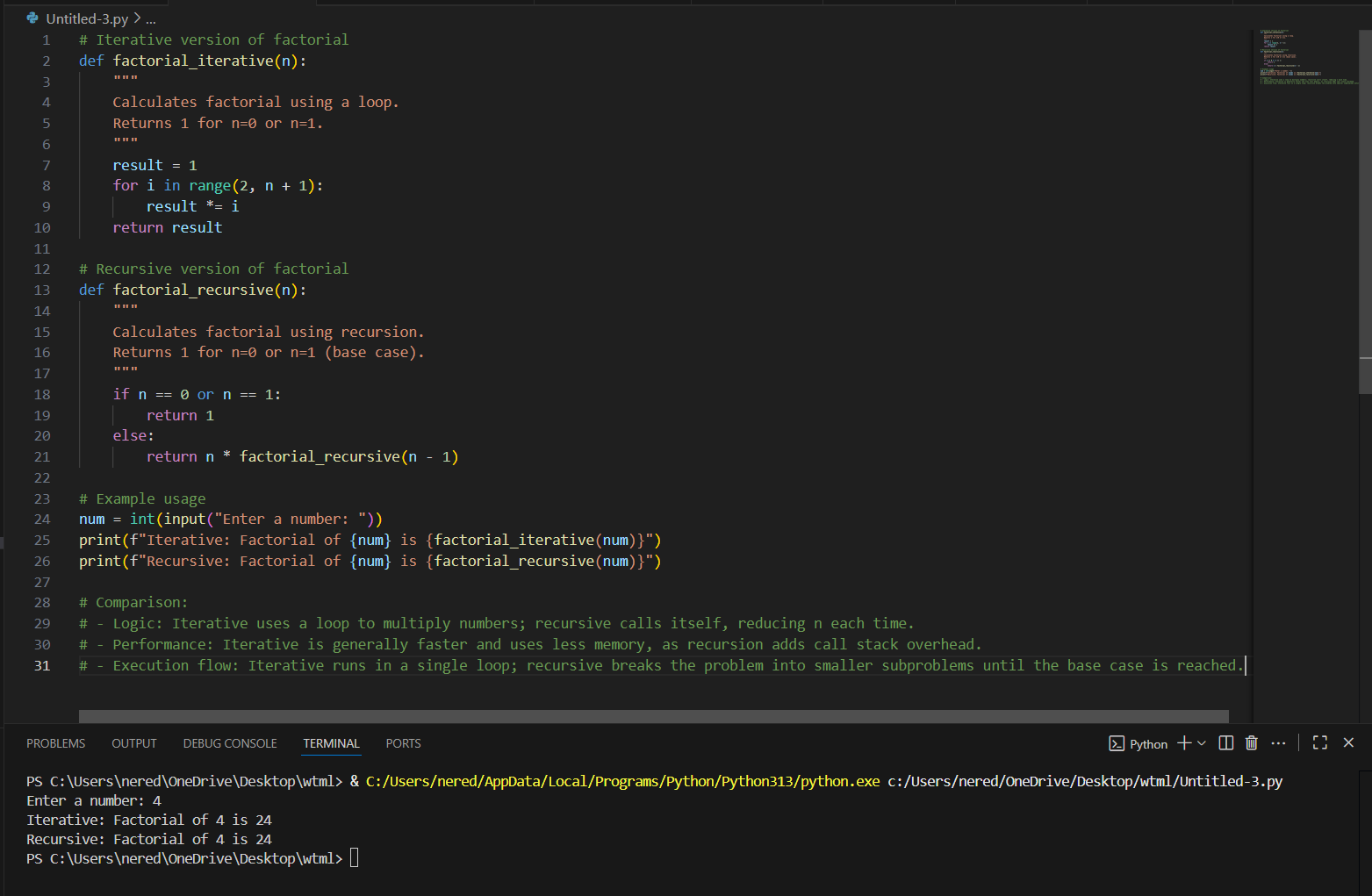
Prompt GitHub Copilot to generate both iterative and recursive versions of the factorial function.

● Expected Output:

o Two correct implementations.

o A documented comparison of logic, performance, and execution flow between iterative and recursive approaches.

Code (screen shot):



Code explanation:

The code provides two ways to calculate the factorial of a number:

1. **Iterative Version (**[factorial\_iterative](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html" \o ")**)**
   * Uses a loop to multiply numbers from 2 up to [n](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
   * Returns 1 for [n = 0](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) or [n = 1](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html).
   * Efficient in terms of speed and memory.
2. **Recursive Version (**[factorial\_recursive](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html" \o ")**)**
   * Calls itself with [n - 1](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) until it reaches the base case ([n = 0](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) or [n = 1](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html)).
   * Returns 1 for the base case.
   * Less efficient for large [n](vscode-file://vscode-app/c:/Users/nered/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) due to call stack overhead.

**Example usage:**

* Prompts the user for a number.
* Prints the factorial using both methods.

**Comparison:**

* *Logic:* Iterative uses a loop; recursive breaks the problem into smaller subproblems.
* *Performance:* Iterative is faster and uses less memory.
* *Execution flow:* Iterative runs in a single loop; recursive uses multiple function calls until the base case.